What Kinds of Models are Needed for Agent-Oriented Modelling and Simulation?

Kuldar Taveter
The University of Melbourne
Talk outline

- Conceptual frameworks.
- Model-Driven Architecture.
- Agent paradigm and Model-Driven Architecture.
- RAP/AOR viewpoint modelling framework.
- The example of B2B e-commerce.
- The example of intruder detection.
- Summary.
The "triangle" framework

- Functions
- Objects
- Processes
- Data
- Events
- Transactions
The ISA (Zachman) framework.

<table>
<thead>
<tr>
<th>Scope</th>
<th>Data (What?)</th>
<th>Function (How?)</th>
<th>Network (Where?)</th>
<th>Actors (Who?)</th>
<th>Time (When?)</th>
<th>Motivation (Why?)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model of the Business</td>
<td>List of things important to the business</td>
<td>List of processes the business performs</td>
<td>List of locations in which the business operates</td>
<td>List of organization units of the business</td>
<td>List of events significant to the business</td>
<td>List of business goals and strategies</td>
</tr>
<tr>
<td>Model of the Information System</td>
<td>ER-diagram (including m:m, n-ary attributed relationships)</td>
<td>Business process model (process flow diagram)</td>
<td>Logistics network (nodes and links)</td>
<td>Organization chart with roles, skill sets, and authorizations</td>
<td>Business master schedule</td>
<td>Business plan with objectives and strategies</td>
</tr>
<tr>
<td>Technology Model</td>
<td>Data model (1:m relationships, fully normalized)</td>
<td>Data flow diagram; application architecture</td>
<td>Distributed system architecture</td>
<td>Human interface architecture (roles, data, access)</td>
<td>Dependency diagram, entity life history</td>
<td>Business rules' model</td>
</tr>
<tr>
<td>Components</td>
<td>Data architecture (tables and columns); mapping to legacy data</td>
<td>System design: structure chart, pseudo-code</td>
<td>System architecture (hardware, software types)</td>
<td>User interface (how the system will behave); security design</td>
<td>&quot;Control flow&quot; diagram (control structure)</td>
<td>Business rules' design</td>
</tr>
<tr>
<td>Functioning System</td>
<td>Converted data</td>
<td>Executable programs</td>
<td>Communication facilities</td>
<td>Trained people</td>
<td>Timing model</td>
<td>Specification of business rules in program logic</td>
</tr>
<tr>
<td></td>
<td>System design: structure chart, pseudo-code</td>
<td>System architecture (hardware, software types)</td>
<td>User interface (how the system will behave); security design</td>
<td>&quot;Control flow&quot; diagram (control structure)</td>
<td>Business events</td>
<td>Enforced business rules</td>
</tr>
</tbody>
</table>
The Enterprise Model framework
<table>
<thead>
<tr>
<th>Step</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conceptual Domain Modelling</td>
<td>Computation Independent Models (CIM)</td>
</tr>
<tr>
<td>Platform-Independent Computational Design</td>
<td>Platform Independent Models (PIM)</td>
</tr>
<tr>
<td>Platform-Specific Implementation</td>
<td>Platform Specific Models (PSM) and Code</td>
</tr>
</tbody>
</table>
The MDA process
Agents and MDA

- Agent as a:
  - modelling abstraction (agent);
  - computational abstraction;
  - implementation unit.

- Agent paradigm spans all three levels of MDA.
The RAP Viewpoint Modelling Framework

<table>
<thead>
<tr>
<th>Viewpoint models</th>
<th>Viewpoint aspect</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Abstraction level</strong></td>
<td><strong>Interaction (org. + interactional)</strong></td>
</tr>
<tr>
<td>Conceptual Domain Modelling</td>
<td></td>
</tr>
<tr>
<td>Platform-Independent Computational Design</td>
<td></td>
</tr>
<tr>
<td>Platform-Specific Design and Implementation</td>
<td></td>
</tr>
</tbody>
</table>
The RAP Viewpoint Modelling Framework populated by models of ROADMAP/Prometheus

<table>
<thead>
<tr>
<th>Viewpoint models</th>
<th>Viewpoint aspect</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Abstraction level</strong></td>
<td>Interaction (org. + interactional)</td>
</tr>
<tr>
<td><strong>Conceptual Domain Modelling</strong></td>
<td>Role Model, Social Model, Models of Actions and Percepts</td>
</tr>
<tr>
<td><strong>Platform-Independent Computational Design</strong></td>
<td>Environment Model, Knowledge Model</td>
</tr>
<tr>
<td><strong>Platform-Specific Design and Implementation</strong></td>
<td>Goal Model</td>
</tr>
<tr>
<td><strong>Platform-Independent Computational Design</strong></td>
<td>Data/Knowledge Diagrams, Interaction Protocols</td>
</tr>
<tr>
<td><strong>Platform-Specific Design and Implementation</strong></td>
<td>Service Models, Agent Lifecycle Models, Agent Overview Diagrams, Process Specifications</td>
</tr>
<tr>
<td><strong>Platform-Specific Design and Implementation</strong></td>
<td>Event Descriptions</td>
</tr>
<tr>
<td><strong>Platform-Specific Design and Implementation</strong></td>
<td>Knowledge and Data Descriptions</td>
</tr>
<tr>
<td><strong>Platform-Specific Design and Implementation</strong></td>
<td>Capability Descriptors, Plan Event Diagrams</td>
</tr>
</tbody>
</table>
The RAP Viewpoint Modelling Framework populated by models of AORML, ROADMAP and UML

<table>
<thead>
<tr>
<th>Viewpoint models</th>
<th>Viewpoint aspect</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Abstraction level</strong></td>
<td><strong>Interaction</strong> (org. + interactional)</td>
</tr>
<tr>
<td></td>
<td><strong>Information</strong> (inform.)</td>
</tr>
<tr>
<td></td>
<td><strong>Behaviour</strong> (motiv. + func. + behav.)</td>
</tr>
<tr>
<td><strong>Platform-Independent Computational Design</strong></td>
<td>UML Use Case Diagrams, AOR Reaction Frame Diagrams, User Interface Design Models, Security Models</td>
</tr>
<tr>
<td></td>
<td>AOR Agent Diagrams</td>
</tr>
<tr>
<td></td>
<td>AOR Reaction Pattern Diagrams, AOR Internal Activity Diagrams, Object Constraint Language</td>
</tr>
<tr>
<td><strong>Platform-Specific Design and Implementation</strong></td>
<td>UML Deployment Diagrams, UML Class Diagrams, UML Interaction Diagrams</td>
</tr>
<tr>
<td></td>
<td>UML Class Diagrams</td>
</tr>
<tr>
<td></td>
<td>UML Class Diagrams, UML State Machine Diagrams</td>
</tr>
</tbody>
</table>
Comparison with other frameworks

<table>
<thead>
<tr>
<th>Abstraction level</th>
<th>Audience/Stakeholders</th>
<th>Viewpoint Names</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conceptual Domain Modelling</td>
<td>owners/customers, users, domain experts</td>
<td>CIM, Enterprise</td>
</tr>
<tr>
<td>Platform-Independent Computational Design</td>
<td>systems analysts, software architects</td>
<td>PIM, Information + Computational</td>
</tr>
<tr>
<td>Platform-Specific Computational Design and Implementation</td>
<td>programmers, database implementers, system integrators</td>
<td>PSM, Engineering + Technology</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MDA</th>
<th>RM-ODP</th>
<th>Zachman</th>
</tr>
</thead>
<tbody>
<tr>
<td>CIM</td>
<td>Enterprise</td>
<td>Rows 1+2</td>
</tr>
<tr>
<td>PIM</td>
<td>Information + Computational</td>
<td>Row 3</td>
</tr>
<tr>
<td>PSM</td>
<td>Engineering + Technology</td>
<td>Rows 4+5</td>
</tr>
</tbody>
</table>
Steps of the domain modelling phase of the RAP/AOR methodology

- Problem Scenario
- Stage I: Identify Actor Types and Relationships
  - Organization Model
    - Stage II: Define Common Knowledge for actors
      - Information Model
    - Stage III: Define Interaction Between Actors
      - Interaction Model
  - Stage IV: Define Behaviour of Actors
    - Agent Behaviour Model
The application areas

- B2B e-commerce.
- Intelligent home.
- Simulation of a ceramics factory.
- „Air-to-air“ business processes.
Simulation of a ceramics factory
Agent-based automation of B2B processes

- Companies participating in a business process (e.g., quoting) are represented by agents. Agents of some companies are “outsourced” to the operator. Circle: agent; arrow: communication; VKB: agent’s knowledge base which includes interfaces to the internal information systems of the company:
Steps of the domain modelling phase of the RAP/AOR methodology

- **Stage I: Identify Actor Types and Relationships**
  - Problem Scenario
  - Organization Model
  - Information Model
- **Stage II: Define Common Knowledge for actors**
  - Stage III: Define Interaction Between Actors
  - Interaction Model
- **Stage IV: Define Behaviour of Actors**
  - Agent Behaviour Model
Organization and interaction modelling (Stages 1 and 2)

- The analysis and modelling of *active entities*, i.e. of agent types and instances and relationships, as well as of *interactions* and *communication*, between them.
AOR Agent Diagram
UML Use Case Diagram

Buyer

Seller

Quote
Order
Pay
AOR Interaction Frame Diagram
AOR Interaction Sequence Diagram

1. request inform RFO/Quote
   globalProductIdentifier = "1247"

2. inform RFO/Quote

3. request provideProduct
   PurchaseOrder/Confirmation

4. inform PurchaseOrder/Confirmation

5. provideProduct
   (PurchaseOrder)

6. request payForProduct
   Invoice

7. payForProduct
   (Invoice)
Information modelling (Stage 3)

- Creating a *domain ontology* which provides a common framework of knowledge for the agents of the organization(s) and external agents connected to the organization(s).
Function and behaviour modelling (Stage 4)

- The modelling of an agent’s functionality and motivation (*what* functions and *why* the agent has to perform), as well as of the agent’s behaviour (*when*, *how* and *under what conditions* work has to be done).

- Function models are missing!
AOR Activity Diagram (incomplete)

- Buyer
  - request inform RFQ/Quote
  - inform RFQ/Quote

- Seller
  - R1
    - Manage quoting (q : Quote)
      - q.quoteLineItem forall -> (quoteLineItemStatusCode.isBid or quoteLineItemStatusCode.isNoBid)
    - Process product items (q : Quote)
    - Confirm quote (q : Quote)
AOR Activity Diagram (complete)
Behavioural patterns

Parallel While-Repeat

Activity Type 1
  ↓
Activity Type 2
  ↓
R1
  {expression1}
  ↓
Activity Type 3
Activity Type 4
  ↓
Activity Type 5

Parallel Repeat-Until

Activity Type 1
  ↓
Activity Type 2
  ↓
R1
  {expression1}
predicate1
  ↓
Activity Type 3
Activity Type 4
  ↓
R2
  {expression1}
predicate1
  ↓
Activity Type 5
## Comparison of modelling methods

<table>
<thead>
<tr>
<th>Pattern</th>
<th>Method</th>
<th>XPDL</th>
<th>UML</th>
<th>BPEL</th>
<th>XLANG</th>
<th>WSFL</th>
<th>BPML</th>
<th>WSCI</th>
<th>AORML</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequence</td>
<td></td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Parallel Split</td>
<td></td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Synchronization</td>
<td></td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Exclusive Choice</td>
<td></td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Simple Merge</td>
<td></td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Multi Choice</td>
<td></td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>Multi Merge</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+/-</td>
<td>+/-</td>
<td>+</td>
<td></td>
</tr>
<tr>
<td>Discriminator</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+/-</td>
<td></td>
</tr>
<tr>
<td>Synchronizing Merge</td>
<td></td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Arbitrary Cycles</td>
<td></td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>MI with a Priori Design Time Knowledge</td>
<td></td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>MI with a Priori Runtime Knowledge</td>
<td></td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>MI without a Priori Runtime Knowledge</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>Deferred Choice</td>
<td></td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Interleaved Parallel Routing</td>
<td></td>
<td>-</td>
<td>-</td>
<td>+/-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
Viewpoints of design

- The internal (*subjective*) perspective of each agent to be implemented is adopted, in contrast to the external (*objective*) perspective adopted in a conceptual domain model.
AOR Reaction Frame Diagram

SellerAgent

BuyerAgent

- request inform RFQ/Quote
- inform RFQ/Quote

- request provideProduct PurchaseOrder/Confirmation
- inform PurchaseOrder/Confirmation

- provideProduct (PurchaseOrder)
- provideProduct (PurchaseOrder)

- request payForProduct Invoice
- payForProduct (Invoice)
- payForProduct (Invoice)
Tools
Alternative approach to implementation

- Executable AOR domain models are transformed into equivalent XML-based subjective representations that are then interpreted and executed by software agents.
- Advantage: design once, implement multiple times!
XML-schema for representing business process types (based on RuleML)

```xml
<xs:element name="businessProcess" type="businessProcessType"/>
<xs:complexType name="businessProcessType">
  <xs:sequence>
    <xs:element name="processTypeName" type="nameType"/>
    <xs:element name="perspective" type="nameType"/>
    <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:element name="reactionRule" type="reactionRuleType"/>
    </xs:choice>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="reactionRuleType">
  <xs:all>
    <xs:element name="ruleName" type="nameType" minOccurs="0"/>
    <xs:element name="event" type="eventTermType"/>
    <xs:element name="body" type="bodyTermType" minOccurs="0"/>
    <xs:element name="head" type="actionEffectTermType"/>
  </xs:all>
</xs:complexType>
```
Schema-based representation of a business process type

<reactionRule>
  <event>
    <startOfActivity>
      <activityTypeName>Process line items</activityTypeName>
    </startOfActivity>
  </event>

  <body>
    <forEach>
      <entityTypeName>QuoteLineItem</entityTypeName>
      <scope>
        <Var>quote</Var>
      </scope>
    </forEach>
  </body>

  <head>
    <mainAction>
      <startActivity>
        <activityTypeName>Process line item</activityTypeName>
        <Var type="QuoteLineItem"/> item
        <activityMode>sequential</activityMode>
      </startActivity>
    </mainAction>
  </head>

</reactionRule>
Buyer Agent

active RFQs

<table>
<thead>
<tr>
<th>product name</th>
<th>product code</th>
<th>valid until</th>
<th>number of responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIKEM 35 PUNANEN JOHDI...</td>
<td>4826</td>
<td>30.03.2004 00:00</td>
<td>1</td>
</tr>
<tr>
<td>MIKEM 1,5 HARVAA JOHDIIN...</td>
<td>4701</td>
<td>30.03.2004 00:00</td>
<td>1</td>
</tr>
<tr>
<td>MIKEM 35 PUNANEN JOHDI...</td>
<td>4826</td>
<td>30.03.2004 00:00</td>
<td>1</td>
</tr>
<tr>
<td>MIKEM 6 PUN ROAST JOHDI...</td>
<td>4766</td>
<td>30.03.2004 00:00</td>
<td>0</td>
</tr>
<tr>
<td>MIKEM 35 PUNANEN JOHDI...</td>
<td>4826</td>
<td>30.03.2004 00:00</td>
<td>0</td>
</tr>
<tr>
<td>MIKEM 2,5 HARVAA JOHDIIN...</td>
<td>4721</td>
<td>30.03.2004 00:00</td>
<td>0</td>
</tr>
<tr>
<td>MIKEM 35 PUNANEN JOHDI...</td>
<td>4826</td>
<td>30.03.2004 00:00</td>
<td>0</td>
</tr>
<tr>
<td>MIKEM 2,5 HARVAA JOHDIIN...</td>
<td>4721</td>
<td>30.03.2004 00:00</td>
<td>0</td>
</tr>
<tr>
<td>SAPØ, UUSA ERIK TYÄKKA...</td>
<td>5036</td>
<td>30.03.2004 00:00</td>
<td>0</td>
</tr>
<tr>
<td>MIKEM 35 PUNANEN JOHDI...</td>
<td>4826</td>
<td>30.03.2004 00:00</td>
<td>0</td>
</tr>
<tr>
<td>SAP Ø LIIIS ERIK TYÄKKA...</td>
<td>5000</td>
<td>30.03.2004 00:00</td>
<td>0</td>
</tr>
</tbody>
</table>

product description

Kytkeänjohto

valid from -> until

31.03.2004 00:00

ordered amount ordered delivery date

34.0 31.03.2004 00:00

issued by (contact info)

Maija Meikäläinen

2 responses received and ranked

<table>
<thead>
<tr>
<th>bidder name</th>
<th>bidder credit</th>
<th>product name</th>
<th>amount</th>
<th>price</th>
<th>delivery date</th>
<th>substitution</th>
<th>status</th>
</tr>
</thead>
<tbody>
<tr>
<td>IEKY</td>
<td>0.99</td>
<td>MIKEM 2,5 HA...</td>
<td>34.0</td>
<td>0.6</td>
<td>31.03.2004 0...</td>
<td>IsbID</td>
<td></td>
</tr>
<tr>
<td>HEDETC</td>
<td>0.7</td>
<td>MIKEM 2,5 HA...</td>
<td>34.0</td>
<td>0.6</td>
<td>31.03.2004 0...</td>
<td>IsbID</td>
<td></td>
</tr>
</tbody>
</table>
The scenario of intruder handling (slide by Leon Sterling)

- Computer vision in a networked home finds a stranger in the house.
- Mining of family images shows he is not a friend or relative.
- The house owner and his family and scheduled guests are warned to stay away via their smartphone, PDA or PC.
- Police is automatically notified, and mining of the police image database brings up the records of the suspect.
Function modelling
(original slide by Leon Sterling)
Implementation I

Activity diagrams of AORML can be straightforwardly transformed into the corresponding JADE constructs and executed in JADE:

- agent types are mapped to JADE agent types;
- object types are mapped to JADE object types;
- activity types are mapped to behaviour types of JADE;
- reaction rules are mapped to the constructs of JADE that invoke and sequence behaviours.
<table>
<thead>
<tr>
<th>Notion of RAP/AOR</th>
<th>Object class in JADE</th>
<th>Object method of JADE (if applicable)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object type</td>
<td>java.lang.Object</td>
<td>-</td>
</tr>
<tr>
<td>Agent type</td>
<td>jade.core.Agent</td>
<td>-</td>
</tr>
<tr>
<td>Elementary activity type</td>
<td>jade.core.behaviours. OneShotBehaviour</td>
<td>-</td>
</tr>
<tr>
<td>Sequential activity type</td>
<td>jade.core.behaviours. SequentialBehaviour</td>
<td>-</td>
</tr>
<tr>
<td>Parallel activity type</td>
<td>jade.core.behaviours. ParallelBehaviour</td>
<td>-</td>
</tr>
<tr>
<td>Execution cycle of a KPMC agent</td>
<td>jade.core.behaviours. CyclicBehaviour</td>
<td>-</td>
</tr>
<tr>
<td>Waiting for a message to be received</td>
<td>jade.core.behaviours. ReceiverBehaviour</td>
<td>-</td>
</tr>
<tr>
<td>Starting the first-level activity</td>
<td>jade.core.Agent</td>
<td>public void addBehaviour (Behaviour b)</td>
</tr>
<tr>
<td>Starting a sub-activity</td>
<td>jade.core.behaviours. SequentialBehaviour</td>
<td>public void addSubBehaviour (Behaviour b)</td>
</tr>
<tr>
<td>Starting a parallel sub-activity</td>
<td>jade.core.behaviours. ParallelBehaviour</td>
<td>public void addSubBehaviour (Behaviour b)</td>
</tr>
<tr>
<td>Start-of-activity activity border event type</td>
<td>jade.core.behaviours. OneShotBehaviour</td>
<td>public abstract void action()</td>
</tr>
<tr>
<td>Start-of-activity activity border event type</td>
<td>jade.core.behaviours. SequentialBehaviour, jade.core.behaviours. ParallelBehaviour</td>
<td>public abstract void onStart()</td>
</tr>
<tr>
<td>End-of-activity activity border event type</td>
<td>jade.core.behaviours.Behaviour</td>
<td>public int onEnd()</td>
</tr>
<tr>
<td>Agent message</td>
<td>java.lang.acl.ACLMessage</td>
<td>-</td>
</tr>
</tbody>
</table>
Simulation
Platform-independent design

Security Agent

ContextGateway

Inform (IntruderDescription)

R1

Handle intruder (cf IntruderDescription)

Identify intruder (cf IntruderDescription)

Inform owner (ic: IntruderDescription)

R2

Receive reply

R3

Respond (ic: IntruderDescription)

Inform police (ic: IntruderDescription)

R4

Inform visitors

Person

isKnown (IntruderDescription)

R5

query-If(isKnown (IntruderDescription))

Inform (no(isKnown (IntruderDescription)))

Communicator Agent

request sendPolice (IntruderDescription)
Summary

- We need:
  - organization models;
  - interaction models;
  - information models,
  - function models;
  - behaviour models.

- It should be possible to describe problems at the level of conceptual domain modelling.

- Tasks of planning, coordination and learning should be “raised” to the domain level.

- Transformation rules between levels should be fully elaborated.

- Is all this realistic?
Research agenda

- Further experience from using RAP/AOR ("air-to-air" business processes).
- Study of the ontological foundations of methodologies (ROADMAP, Prometheus) in cooperation with Giancarlo, Gerd, Leon and others.
- Research work on exception handling in cooperation with Renata, Gerd and others.